1999-1

A Feature Library Approach to On-line Image Querying and Retrieval for Topographic Applications

James Carswell

Technological University Dublin, jcarswell@tudublin.ie

1999-01-01

A feature library approach to on-line image querying and retrieval for topographic applications

James D. Carswell
Dublin Institute of Technology, jcarswell@dit.ie

A Feature Library Approach to On-line Image Querying and Retrieval for Topographic Applications

Peggy Agouris James Carswell Anthony Stefanidis

Dept. of Spatial Information Science & Engineering
and the National Center for Geographic Information and Analysis
University of Maine
5711 Boardman Hall #348
Orono, ME 04469-5711, USA
{peggy, carswell, tony}@spatial.maine.edu

Abstract

In this paper we address the problem of content-based image retrieval using queries on shape and topology. We focus on the particularities of image databases encountered in typical topographic applications, and present the development of a spatial data management system that enables such queries. The query uses as input user-provided sketches of the shape and spatial configuration of the object (or objects) which should appear in the images to be retrieved. The objective of the search is to retrieve images that contain a configuration of objects sufficiently similar to the one specified in the query. Our novel approach introduces the design of a structured feature library which is linked to an integrated image database in addition to the development of the necessary matching tools. We discuss our overall strategy and focus on the use of the feature library to support our queries.

1. INTRODUCTION

The intelligent retrieval of images from large databases is the focus of substantial research efforts within the computer vision community [Pickard and Minka, 1995; Sclaroff et al., 1997; Ogle and Stonebraker, 1995; Gudivada and Ragavan, 1995; Jain, 1992; Cohen & Guibas, 1996; Gupta et al., 1998]. The objective is to retrieve specific images from a large database by querying on properties of these images. As a result of pioneering research efforts, some prototype systems have been reported, with few notable examples being Virage [Gupta, 1995], Chabot [Ogle and Stonebraker, 1995], IBM's QBIC [Flickner et al., 1995], VisualSeek [Smith and Chang, 1995], ImageRover [Sclaroff et al., 1997], and PicHunter [Cox et al., 1997].

Most of these efforts address the problem within the context of multimedia applications, and therefore they focus on general-use, multimedia-type image databases. In such applications, low-level image properties (e.g. color and texture) are often adequate for information retrieval, since the image members of the database display substantial differences in these properties and can be distinguished by them alone. However, in a variety of applications we have databases containing large numbers of images which are often extremely similar in terms of general low-level image properties. Databases of aerial and satellite image databases are one such example. Therefore, general-purpose image retrieval approaches like the ones mentioned above are not sufficient for information retrieval in topographic image databases. Instead, what distinguishes images in a topographic database is their actual content: the *shape* and *configuration* of the objects they contain.

In this paper we present the strategy and design considerations behind ***I.Q.*** (Image Query by Sketch), our prototype system for image retrieval (section 2), emphasize on the role and organization of feature libraries for image retrieval (section 3), and discuss some digital image analysis issues related to matching sketches to images for querying (section 4). It should be mentioned that while our research originates from topographic applications, the developed methodology can be applied to any type of imagery.

2. STRATEGY AND SYSTEM DESIGN

A description of the operation environment for our system is shown in Fig. 1. Our searchable topographic database comprises images (typically aerial or satellite), outline (edge) information for physical entities depicted in these images, and metadata. Metadata of interest include the typical information (e.g. sensor characteristics, date of capture, scale) which describes and enhances the content and/or properties of common topographic data files (e.g. digital images, digital elevation models, maps in digital format).

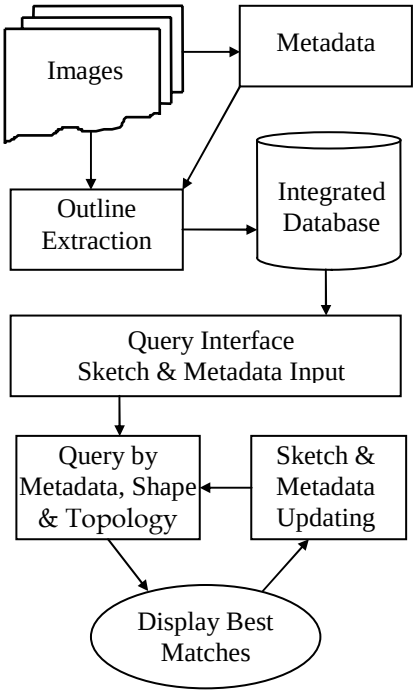

Figure 1. Operation environment for ***I.Q.***

The aim of our strategy is to take advantage of the intuitive method humans use to express spatial scenes, namely *sketching*. In accordance, the query interface of ***I.Q.*** intends to allow the user access to individual members of this database by using *metadata* information and *outlines* as input parameters. The operator sketches a configuration of objects, provides additional metadata information, and the database is searched to yield the images which satisfy the given metadata information, and from these, the images in which spatial configurations appear similar to the given sketch. In order to support our queries, our searchable database comprises three components:

Image library: contains one entry for every image of the database, and provides a link/pointer to the features contained within them an stored in the Feature Library.

Metadata library: contains a listing of potential values for a set of attributes that describe general properties of the images. These attributes include date and time of acquisition, date and time of introduction in the database, scale/resolution, and location of the image (expressed in hierarchically arranged geographic entities like state, county, city). For more complex databases the attributes may be extended to incorporate information on the sensor and imagery type (e.g. b/w, color, or pseudocolor).

Feature library: contains a set of distinct features (i.e. image object shapes) and links to image files where such features appear. The role of the feature library is to provide the crucial link (that allows us to reduce the search space of a query) from a large image database to an abridged group of features, thus allowing for on-line querying.

Under this design, during the *on-line* part of a query, the input object outlines are matched to elements of the feature library using an on-line matching tool, and acceptable matches give links to specific images (and locations within them). Our matching tool is based on a modification of least squares matching (lsm) to function using object outlines (edges) as input. The input outline is allowed to deform through an affine transformation (two translations, rotations, and scalings) to match a library feature. A matching percentage expresses the quality of the match, corresponding to the percentage of perfect pixel correspondences between the input outline and the library entry. For a more detailed description of the matching tool, please refer to [Agouris et al., 1998]. An *off-line* matching process is performed to establish links between feature library entries and image locations. A detailed description of the off-line and on-line processes may be found in [Agouris et al, 1998b].

3. THE FEATURE LIBRARY - STRUCTURE & ORGANIZATION

The feature library permits us to narrow the search space from a large database of images to a limited group of feature outlines. In order for the query to be efficient, the library needs to be organized in an optimal way. The optimality criteria are two: the members of the library should be *exhaustive* (thus being able to describe all possible query input features), and the members of the library should be *independent* (minimizing unnecessary duplications). The two properties, when satisfied, are equivalent to an ideal library, which is approaching a base spanning the space of shapes.

The organization of data within the feature library is intended to enable it to act like a multi-stage screening mechanism that minimizes the risk of wasting considerable time making passes over extensive data that have no

chance of selection. For example, the first screening criterion will eliminate as potential matching candidates most of the features within the library. The secondary screening criterion will eliminate the next greatest number of alternatives and so on down through the feature library tree hierarchy.

Furthermore, due to the dynamic natures of the image database and query processing, the feature library is constantly adding, subtracting and otherwise updating its features, links, and internal organization. It therefore needs to be autonomous in that it be able to automatically maintain its contents depending on the changing states of these external but integrated components. The feature library is organized according to a hierarchical (tree-like) structure (Fig. 2) as follows:

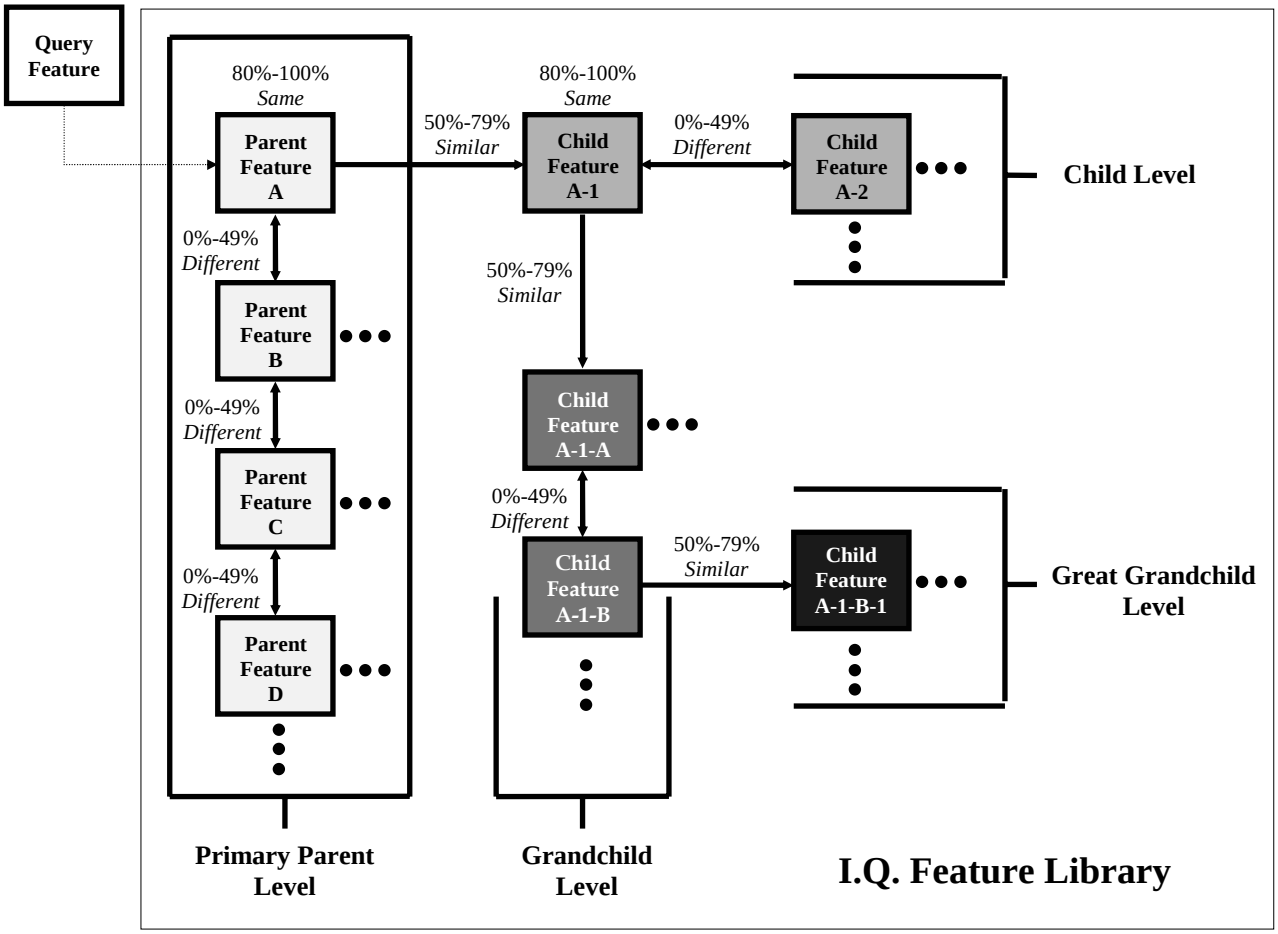

Figure 2 - Feature Library Hierarchy

The Primary Parent Level - where every new query shape is first tested against, and its respective matching percentage recorded. The features at this level are the roots of all trees comprising the feature library. Different features within the parent level are considered dissimilar, as their mutual similarity percentages are lower than 50%. When a query is performed, the matching percentage between the query feature and a parent feature will fall in one of three ranges: (0%-49%), (50%-79%) and (80%-100%). In the latter case, i.e. (80%-100%) range, the query feature is considered to be the "same" as the parent feature. The links the parent feature has to the images in the image database are then returned as the results to the query. In the event that there exist more than one parent feature that matches 80% to 100% to the query feature, the links of the highest matching parent are returned first, with the links of the remaining matches following in order. In practice, this case of multiple parent matches is rare, due to the mutual dissimilarity of the parent features. For the (0%-49%) range, the query feature is considered to be "different" than the parent feature. If this holds for all parent level features, the query feature is a candidate to be inserted into the feature library, at the primary parent level.

Subsequent off-line processing is then required to establish its links to images in the database .

The Child Level - If the query feature matched in the (50%-79%) range to any parent feature, it is considered as "similar". It then tests against this parent's respective child features. In case of multiple parent candidates, we select the one with the highest percentage first and continue with other candidates in order. If there are as yet no child features for the selected parent, the query feature gets inserted into the feature library as a descendent of the best matched parent. Its links are then added and prioritized through off-line matching. If there are child features already residing at this level for any of the selected parents, the query feature gets matched against each of them. If the matching percentage is in the (80%-100%) range, the query feature is considered the "same" as this child feature. The links of this child feature to the image database are returned as the results to the query. If the matching percentage is in the (0%-49%) range, then the query feature gets inserted into the feature library at the child level of the best matched parent, and additional links are added and prioritized through off-line matching.

The Grandchild Level – If, after matching in the (50%-79%) range to a parent feature, the query feature matched in the (50%-79%) range to any child feature, the query feature is tested against the respective grandchild features. Obviously, the relationship between child and grandchild is similar to the one between parent and child. Similar processes to the ones already described take place at this library level to identify the "same" grandchild, to establish a new grandchild, or to move further down the tree to the level of great-grandchildren. Combined, these similarity tests, that have to be satisfied within branches of the tree structure, form the *insertion testing* criterion.

4. FEATURE LIBRARY HOUSECLEANING

The method described above, whereby new features are compared to existing ones and progress through tree levels until they are eventually matched to an entry or inserted in the library might result in some inconsistencies. For example, it is possible that a new query feature might match above 50% to a particular child/sub-child feature but less than 50% to any of the current parent features. This results in the query feature being inserted into the feature library at the primary parent level although the child/sub-child feature that matched better than 50% to this newly inserted parent feature might now be residing under the "wrong" parent. In this case the child feature must shift its position within the feature library to reside under the "better" parent in the tree hierarchy.

Another example of a potential inconsistency in the feature library could occur when two sub-children from different primary parents match closer together than to any other feature currently within their respective tree hierarchies. Depending on the temporal sequence of feature insertion, this phenomenon could produce unnecessary duplicates in some cases (Fig. 3a) and necessary duplicates in others (Fig. 3b). This is due to the allowable range of percentages considered for "same" (i.e. 80% or above) and "similar" (i.e. 50% to 79%) matching. If the feature library did not allow for this degree of uncertainty in its matching by allowing for exact matching only, this phenomenon would not exist. These, and other potential inconsistencies are rectified by applying general feature "housecleaning" (which utilizes the temporal aspect of feature insertion) to the feature library. This process is performed off-line, to ensure the continuous proper organization.

Feature library housecleaning is a process that corrects inconsistencies of the feature library tree hierarchy and is run by default at regular intervals (e.g. at the end of a session, or every time a preset number of new features has been inserted in the library). It ensures that all child/sub-child features are residing under their proper parent and minimizes the theoretical possibility that unnecessary duplicate features could exist somewhere in the tree hierarchy (i.e. child/sub-child features that are between 80% and 100% similar). It also ensures that each of the parents are themselves unique.

Feature housecleaning utilizes a self-organizing table called the "Temporal Feature Index" (TFI). The TFI is a two dimensional cross referencing table of feature filenames together with their respective matching percentages. Features inserted into the feature library are simply appended to this table in the order that they are introduced in the library, and all their matching percentages recorded as they occur. Thus, ordering within TFI reflects relative insertion time. The time of insertion is important so as to reduce both the number of features to test and the number of features to test against. For example, it is not necessary to re-test Child Feature A-2-A against Parent Feature A as this matching percentage is already known and recorded previously at time of insertion.

Feature housecleaning searches the TFI top-down beginning with the first feature. It takes this feature and matches it against all other features inserted after it, provided there does not already exist a matching percentage for it in the table. It should be noted that matching percentages do not hold a reflective property. The percentage of match between feature X and Y is not necessarily the same as between Y and X. An obvious example is the case where one of the two is a more complex structure, containing completely the other one.

If the matching percentage in the above mentioned comparison is in the (80%-100%) range to any of the subsequently added features, feature A will want to remove itself from the feature library. If the matched feature is a parent level feature, the links for feature A get passed to this "same" parent feature and then deletes itself from the library. If the matched feature is not a parent feature, then the matching percentages for all the nodal features above this match beginning at the primary parent level are tested, moving up the tree. If all pass the 50% to 79% similarity test, the feature is an unnecessary duplicate and therefore passes its links to this "same" feature and removes itself from the feature library. If the parent/child feature had sub-children of its own, they would momentarily be left "parentless" but are tested in turn similarly to their shifting parent as they resided after it in the TFI.

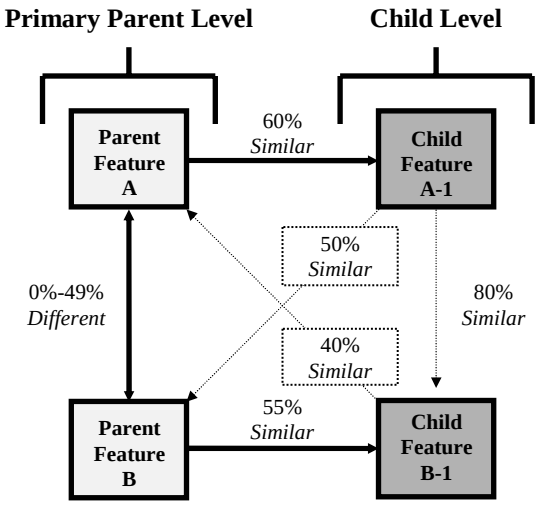

Figure 3a - Unnecessary Duplications.
Parent Features A and B already exist in the library. Child Feature A-1 matches 60% to A and 50% to B. It therefore gets correctly inserted as Child Feature A-1. Child Feature B-1 matches 40% to A and 55% to B so gets correctly inserted as Child Feature B-1 without testing against A-1. Child Feature A-1 is 80% similar to B-1 and 50% similar to B so should pass its links to B-1 and be removed from the feature library.

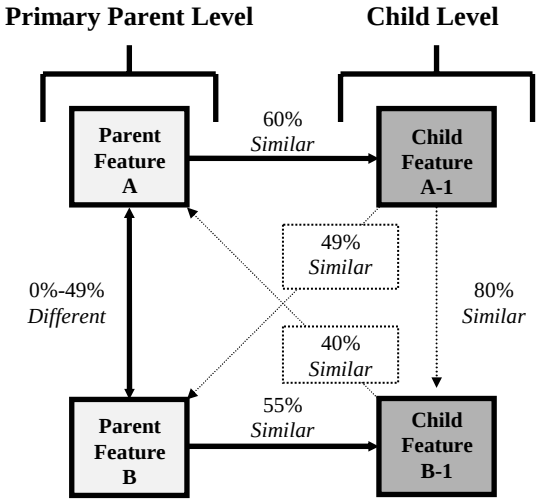

Figure 3b - Necessary Duplications.
Parent Features A and B already exist in the library. Child Feature A-1 matches 60% to A and 49% to B. It therefore gets correctly inserted as Child Feature A-1. Child Feature B-1 matches 40% to A and 55% to B so gets correctly inserted as Child Feature B-1 without testing against A-1. Child Feature A-1 is 80% similar to B-1 but 49% similar to B so is not allowed to shift its position under B in the tree hierarchy - thus maintaining library consistency.

If the initial matching percentage is between 50% and 79% to any of the subsequently added features and at the same time better than the current matching percentage to its immediate parent, the insertion testing criterion is applied to all nodal features beginning with the primary parent of this "better" matched feature. If all insertion tests are satisfied down to the level of the better matched feature, the child feature shifts its position in the tree hierarchy under this new parent - providing it is less than 50% similar to any existing children already occupying this level. This eliminates the possibility of a shifting feature ending up in a position within the tree "worse off" than where it was before (Fig. 4). It also ensures that the internal consistency of the library is maintained, i.e. all features reside under their best matched parent, all things considered (i.e. satisfying the insertion testing criterion). If a feature shifts position within the tree, its position within the TFI is also affected by shifting to the bottom of the table. All features previously below it in the table move up one notch in the index.

Similar to the case mentioned above where unnecessary duplicate features are removed from the library, if the shifting child feature had sub-children of its own, they would momentarily be left "parentless". However, these children are tested in turn similar to their shifting parent as they resided after it in the TFI. It can be envisioned that once tested, these temporarily "parentless" children could conceivably find themselves shifting back under their previous parent in the tree. Feature housecleaning terminates once it has completed one pass through the TFI table.

4. EXPERIMENTS AND COMMENTS

A prototype system of ***I.Q.***, the image query environment described in this paper has been implemented in a Windows workstation. Search and retrieval times for the feature library methodology was tested (using a single 180MhZ Pentium) against a sample database of 64 features. The features provided links to an image database of around 50 files. These features comprised a grouping of both manually sketched and extracted objects from existing imagery. Typical shapes were of generic nature (e.g. circular, representing cooling towers, and rectangular, representing buildings) and unique shapes like outlines of airplanes and other visible image objects.

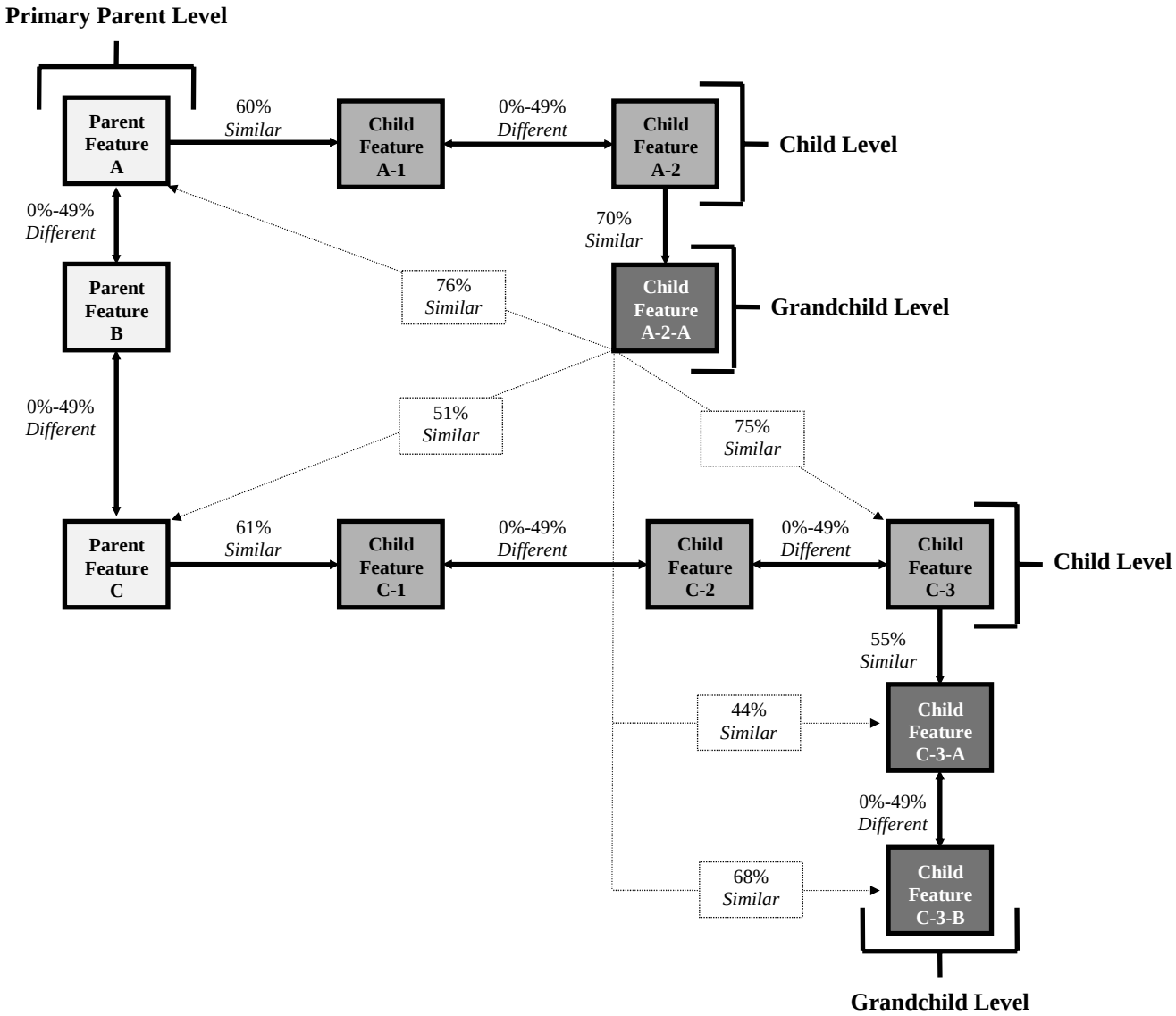

Figure 4 - Feature Housecleaning.
Feature A-2-A matches better to C-3 than to A-2 and should perhaps be a descendent of this "parent". However, because C-3 has a child (C-3-B) that matches above 50% to A-2-A but less than it's current parental match of 70%, it doesn't shift its position within the tree as this would make the feature library inconsistent. Had A-2-A matched above 70% to C-3-B, it would have shifted position under this feature in the tree hierarchy.

Testing the matching algorithm for one feature against a complete but unstructured feature library, i.e. one that has not been organized into a hierarchical tree structure like the one described in sections 4 and 5 of this paper, resulted in search times averaging around 2'20" per database. It takes approximately the same time to match a single feature against a single 512x512 pixel block of aerial image. A substantial savings in search times therefore is realized through matching query sketches to linked features rather than to the original images, which was to be expected.

The next step was to organize the 64 features into their proper tree hierarchy according to the rules outlined previously, using the (80%-100%) range for "same", (50%-79%) for "similar", and (0%-50%) for "different". The first pass through the list of features resulted in a tree that was 38 features wide and up to 3 deep (grandchild level) with 11 features being removed (deleted) as unnecessary duplicates. After applying feature housecleaning, the tree reduced to a structure of 31 features wide and 3 deep, with one additional feature being removed as an unnecessary duplicate. The search times therefore were cut in half as the minimum number of features to test against was reduced from 64 to 31. Additional testing into the tree, if required, did not take significant amounts of time as it was only three features deep, with typical search times averaging around 2" per feature.

These early results support the notion that substantial amounts of search time can be saved if features are organized into a tree hierarchy structure where features with similar shape characteristics are grouped together. Using this ***I.Q.*** Feature Library approach to image database search and retrieval therefore allows for raw imagery to be queried on-line.

Acknowledgments

This work was partially supported by the National Science Foundation through grant number SBR-8810917 for the National Center for Geographic Information and Analysis, and through CAREER grant number IRI-9702233.

References

[1] Agouris P., A. Stefanidis & J. Carswell *"Intelligent Retrieval of Digital Images from Large* Geospatial Databases", International Archives of Photogrammetry and Remote Sensing, Vol. XXXII, Part 3/1, pp. 515-522, 1998, Columbus, OH.

[2] Agouris P., A. Stefanidis & J. Carswell *"Digital Image Retrieval Using Shape Based Queries"*, Spatial Data Handling '98, pp. 613-625, 1998b, Vancouver, Canada.

[3]Cohen S.D. & L.J. Guibas. Shape-Based Indexing and Retrieval; Some First Steps. in *Proceedings 1996 ARPA Image Understanding Workshop*, Vol. 2, pp. 1209-1212, 1996.

[4] Cox I.J., J. Ghosn, M.L. Miller, T. Papathomas & P. Yianilos. Hidden Annotation in Content Based Image Retrieval. *Proceedings IEEE Workshop on Content-Based Access of Image and Video Libraries*, pp. 76-81, 1997.

[5] Flickner M., H. Sawney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkaani, J. Hafner, D. Lee, D. Petkovic, D. Steele, & P. Yanker. Query by Image and Video Content: The QBIC System. *Computer*, Sept. 1995, pp. 23-32, 1995.

[6] Gudivada V. & V. Raghavan. Design and Evaluation of Algorithms for Image Retrieval by Spatial Similarity. *ACM Transactions on Information Systems*, Vol. 13, No. 2, pp. 115-144, 1995.

[7] Gupta A. Visual Information Retrieval: A Virage Perspective. TR95-01, Virage, Inc., San Mateo, CA. 1995.

[8] Gupta A., S. Santini, & R. Jain. In Search of Information in Visual Media, *Communications of the ACM*, Vol. 40, No. 12, pp. 34-42.

[9] Jain R. *NSF Workshop on Visual Information Management Systems.* Redwood, CA, 1992.

[10] Ogle V.E. & M. Stonebraker. Chabot: Retrieval from a Relational Database of Images. *Computer*, pp. 23-32, Sept. 1995.

[11] Pickard R.W. & T.P. Minka. Vision Texture for Annotation. *Multimedia Systems*, 13(1) pp. 3-14, 1995.

[12] Sclaroff S., L. Taycher & M. LaCascia. ImageRover: A Content-Based Image Crowser for thr World Wide Web. *Proceedings IEEE Workshop on Content-Based Access of Image and Video Libraries*, pp. 2-9, 1997.

[13] Smith J.R. & S.-F. Chang. Searching for Images and Video on the World Wide Web. *Multimedia Systems*, Vol3, No. 1, pp. 3-14, 1995.